

Video capture optimization with latency estimation on TV Workstation

Student:

Xuan-Quang Le

Supervisors:

Tifenn RAULT

Mathieu DELALANDRE

Polytech Tours , University of Tours , France

May 31th , 2024

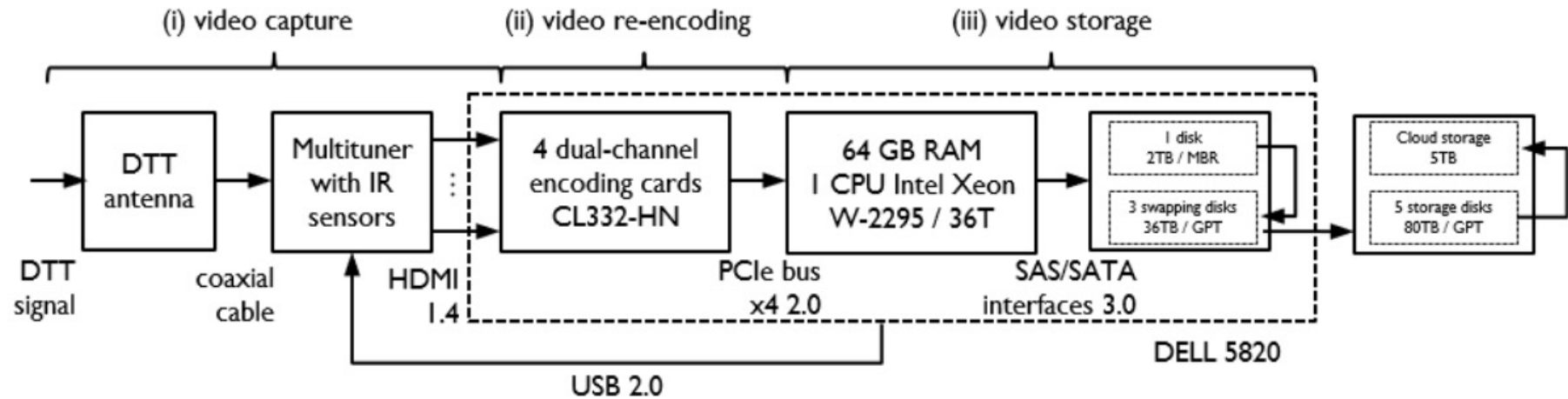


Outline

- 1 TV Workstation
- 2 The runtime for dynamic video capture
- 3 The algorithm to optimize the scheduling
- 4 Conclusions and perspectives

TV Workstation (1/4)

- DELL 5820 computer processes 9 disks and 120 TB of capacity for external storage (HD, 30 FPS, 24h/day), with real-time audio / video encoding, control of tuners with IR sensors

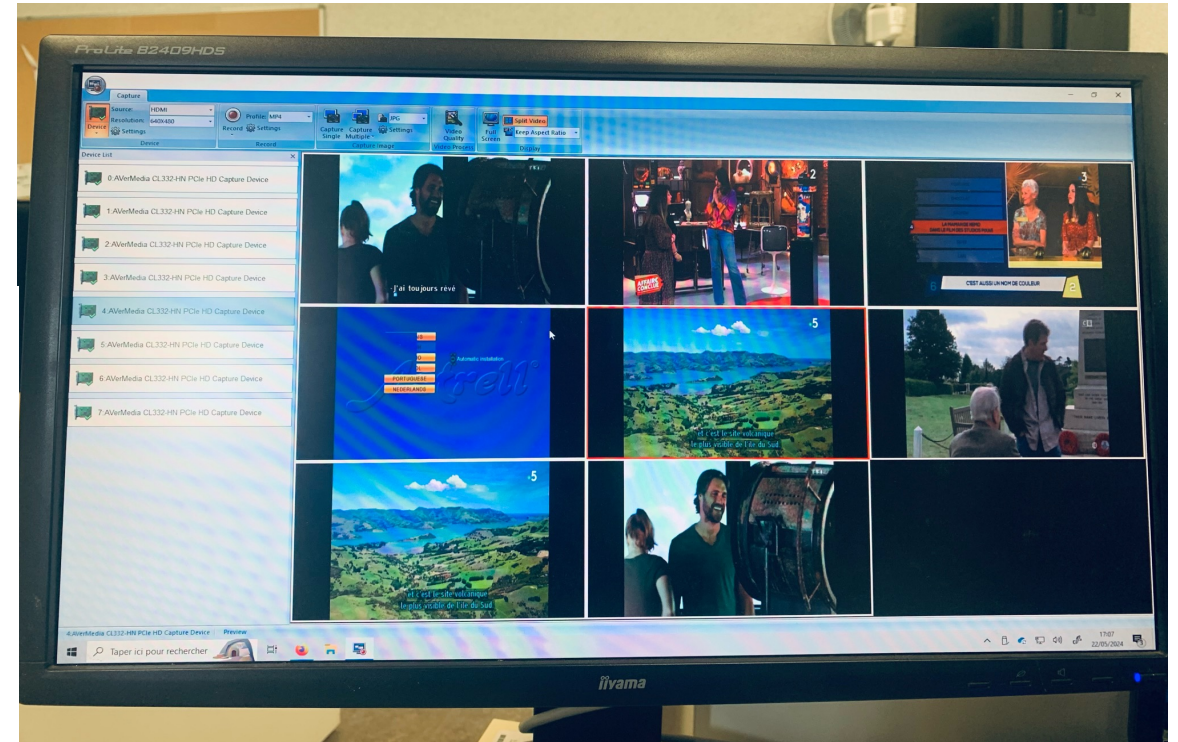
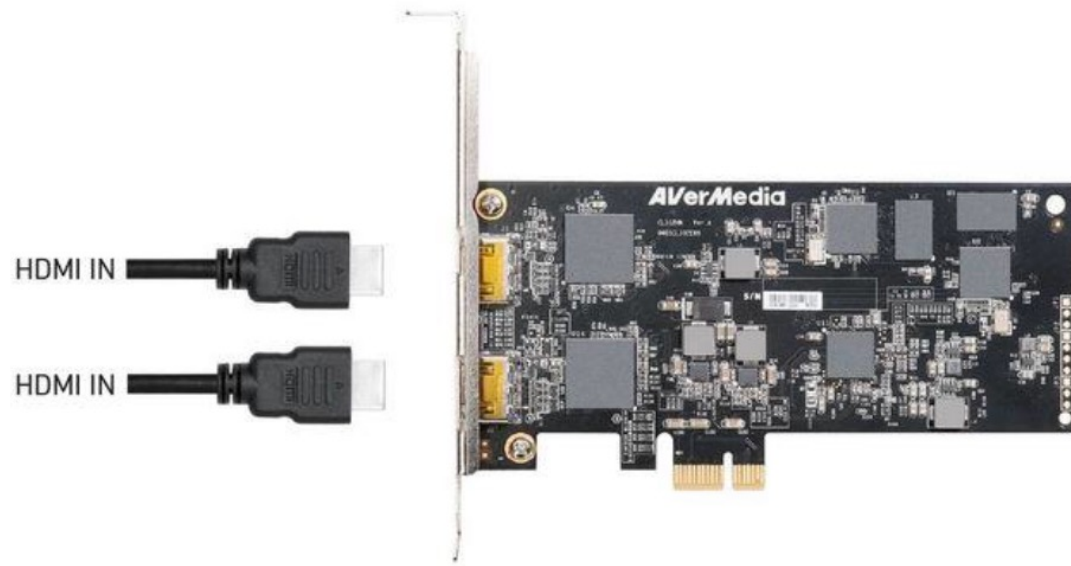


The architecture of the TV Workstation for video capture

TV Workstation (2/4)

➤ Avermedia :

- Support Dual-Channel
- Full-HD capture(1920 x 1080 30fps)
- Hardware H.264 encoding
- Standard HDMI connector



Avermedia CL332-HNPCIe HD capture device

TV Workstation (3/4)

- DELL 5820 computer performance for video storage

Resolution		Audio/ video		Video Mbps	TB/ month		Audio Kbps	GB/ month
HD	1280x720	asyn		3	7.23		256	621
SD	720x576			1.6	3.89		160	384
Low	320x240			0.56	1.36		128	308

TV Workstation (4/4)

➤ we capture along in multiple channels with two modes.

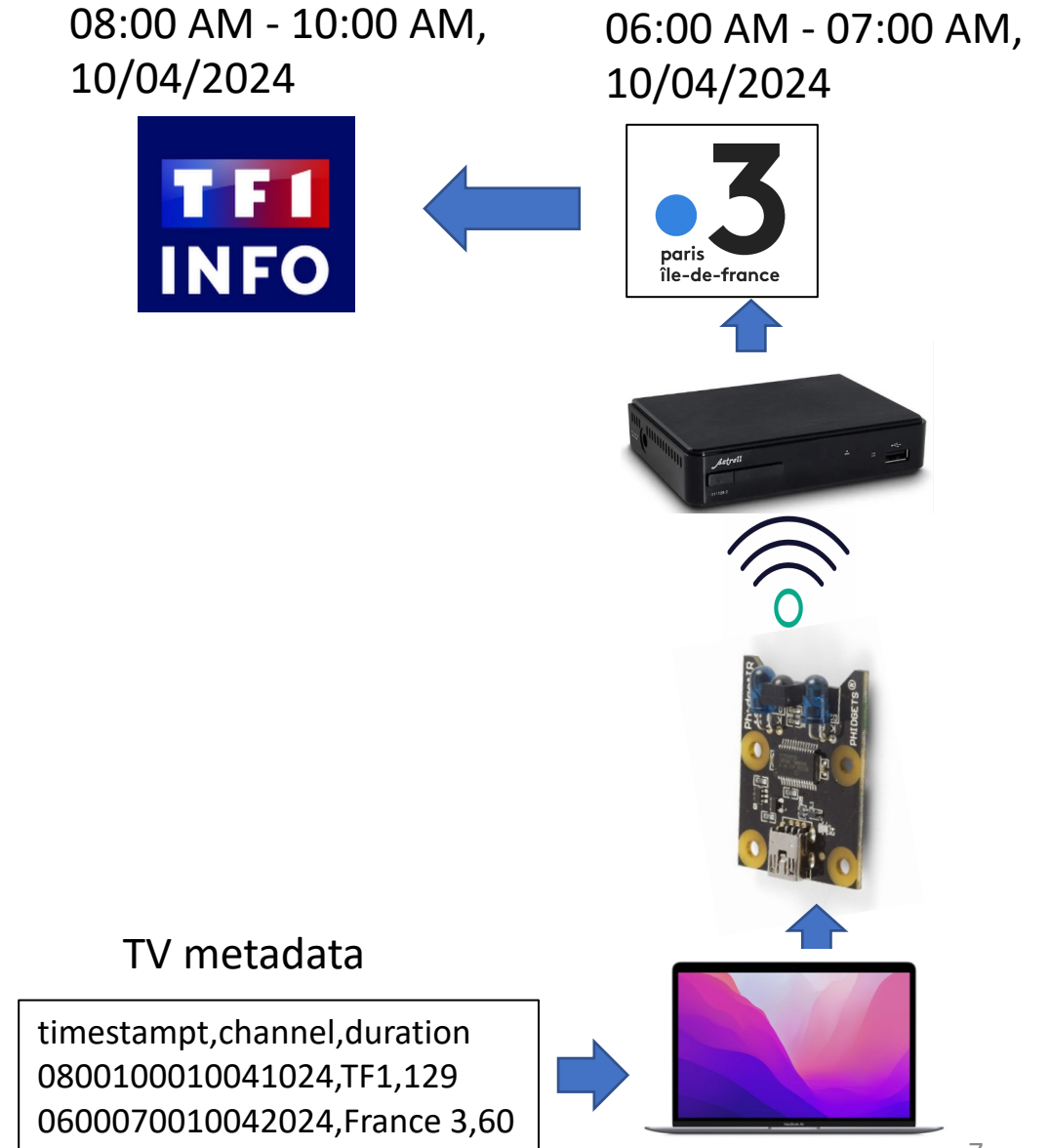
Modes	Details	Pro	Cons
Static and continue capture	<ul style="list-style-type: none">Each input of card is assigned to a single channel with a static assignment (no switch between the channels during the capture).The capture cannot be made idle.	Easy to setup	<ul style="list-style-type: none">No break, require long captures and a huge storageCannot filter the duplicate data (twice daily shows)Can capture only 8 different channels.
Dynamic capture	<ul style="list-style-type: none">Each input of card can be controlled to switch between channels, even made idle for a time between two captures.	<ul style="list-style-type: none">Require short captures with a limited storageCan capture many different channels at one time (>8)Can filter the duplicate data	More difficult to handle

➤ Our solutions:

- A runtime for dynamic capture
- An algorithm to optimize the scheduling

The runtime for dynamic video capture(1/4)

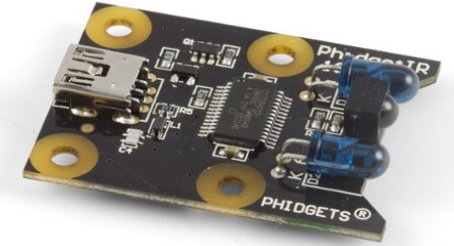
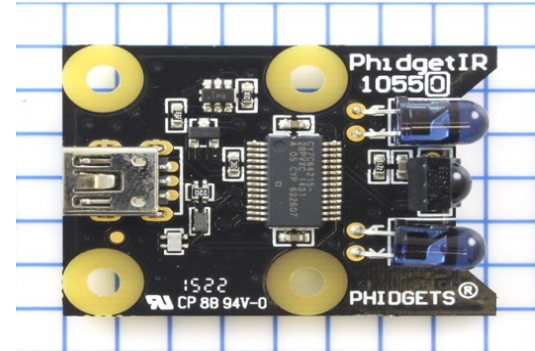
- We read TV schedule (csv file)
- We have developed a program tuning TV channels and Set channel switching time (T)
- Send the channel transmit code from the sensor to the Tuner at that time
- Switch TV channels automatically



The runtime for dynamic video capture(2/4)

➤ Phidgets sensor

- Sensor (phidget IR 1055) : transmit signals
- Hardware mounting kit : fixed position
 - 4x M3 Bolts (2cm Length)
 - 4x Plastic spacers (5mm Length)
 - 4x M3 Nuts
- USB Mini-B : Connect the computer to the sensor

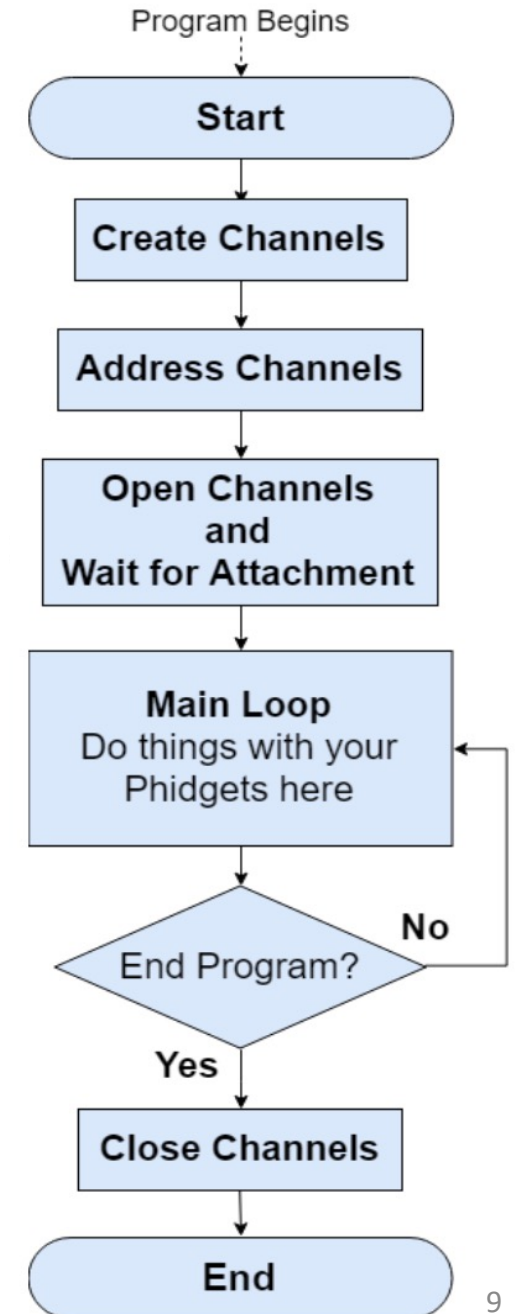


Phidget sensor

The runtime for dynamic video capture (3/4)

➤ Control phidgets sensor :

- Every Phidget channel in any program will follow the same life cycle
- Create channels
- Set some basic parameters to indicate which Phidget to connect
- Main loop : Switch channels ...
- End program



The runtime for dynamic video capture (4/4)

➤ A demo for single recording

- Input: A list of TV programs (illustrated in Tab 1)
- Output: A list of actions (sending a signal to the sensor, starting / stopping the record) correspond to a given timestamp. (shown in Fig. 1)

➤ Supplement steps

- Sorting the TV program list
- Defining the switching time as Equation

➤ Set channel switching time :

$$T = t1 + \frac{(t2-t1)}{2}$$

- T: Switch time
- t1 is the stop_time of the current channel
- t2 is the start_time of the next channel
- Exception : The first channel, the channel switch time will be 1 minute before the start time

➤ The table follow the structure :

ch1; t0; t1

ch2; t2; t3

ch3; t4; t5

....

where $t0 < t1 < t2 \dots < t7$ and $ch1 * ch2 \dots * ch4$ (in a simple way, $ch2 = ch1 + 1$)

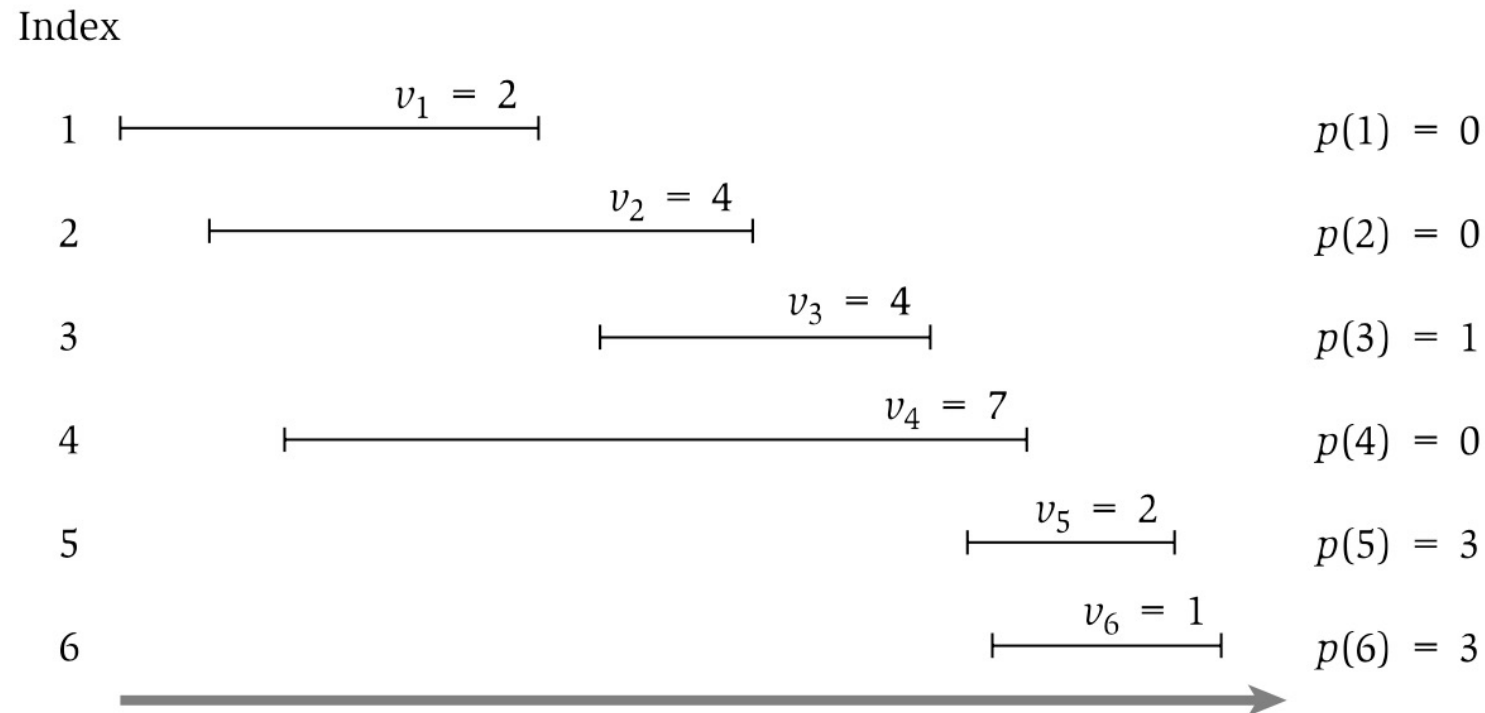
index	channel_name	start_time	stop_time
0	Channel 2	18:58:00	18:59:00
1	Channel 3	19:01:00	19:02:30
2	Channel 5	19:04:45	19:05:00

```
Starting ....
Total rows: 3
Time: 18:58:00
Switching to channel 2 at switch time 2024-05-23 18:57:00
Transmitted code 106F42BD for channel 2
Time: 19:00:00
Currently capturing channel: 3 at switch_time 2024-05-23 19:00:00
Transmitted code 106F827D for channel 3
Time: 19:03:30
Currently capturing channel: 5 at switch_time 2024-05-23 19:03:30
Transmitted code 106F629D for channel 5
```

The algorithm to optimize the scheduling (1/4)

➤ Kleinberg algorithm

- Input: A set of time intervals, each with a value (also called weight).
- Output: Subset of non-overlapping intervals with the largest total value.



An instance of weighted interval scheduling with the functions $p(j)$ defined for each interval j .

The algorithm to optimize the scheduling(2/4)

➤ Kleinberg algorithm

➤ Sort intervals by stop_time :

- the intervals are sorted by their end_time using the quicksort algorithm.

➤ Compute P value(P[j])

➤ Compute the value M:

- M[j] as the maximum value of the weighted sum of non-overlapping time periods when considering interval j .
- Calculation formula:

$$M[j] = \max\{v[j] + M[P[j]], M[j - 1]\}$$

- v[j] is the value (weight) of interval j.
- M[j-1] is the maximum weighted sum for intervals from 0 to j-1
- M[P[j]] is the maximum weighted sum for non-overlapping intervals up to P[j].

➤ Find solution:

- Based on the M and P values, we can trace back to find the subset of time intervals that make up the optimal solution.

The algorithm to optimize the scheduling (3/4)

- We need to "correct" the algorithm.
 - Save the list of beginning(O)
 - Save the list of interval selected (S)
 - Save the list of remaining intervals(R)

$$R = O - S$$

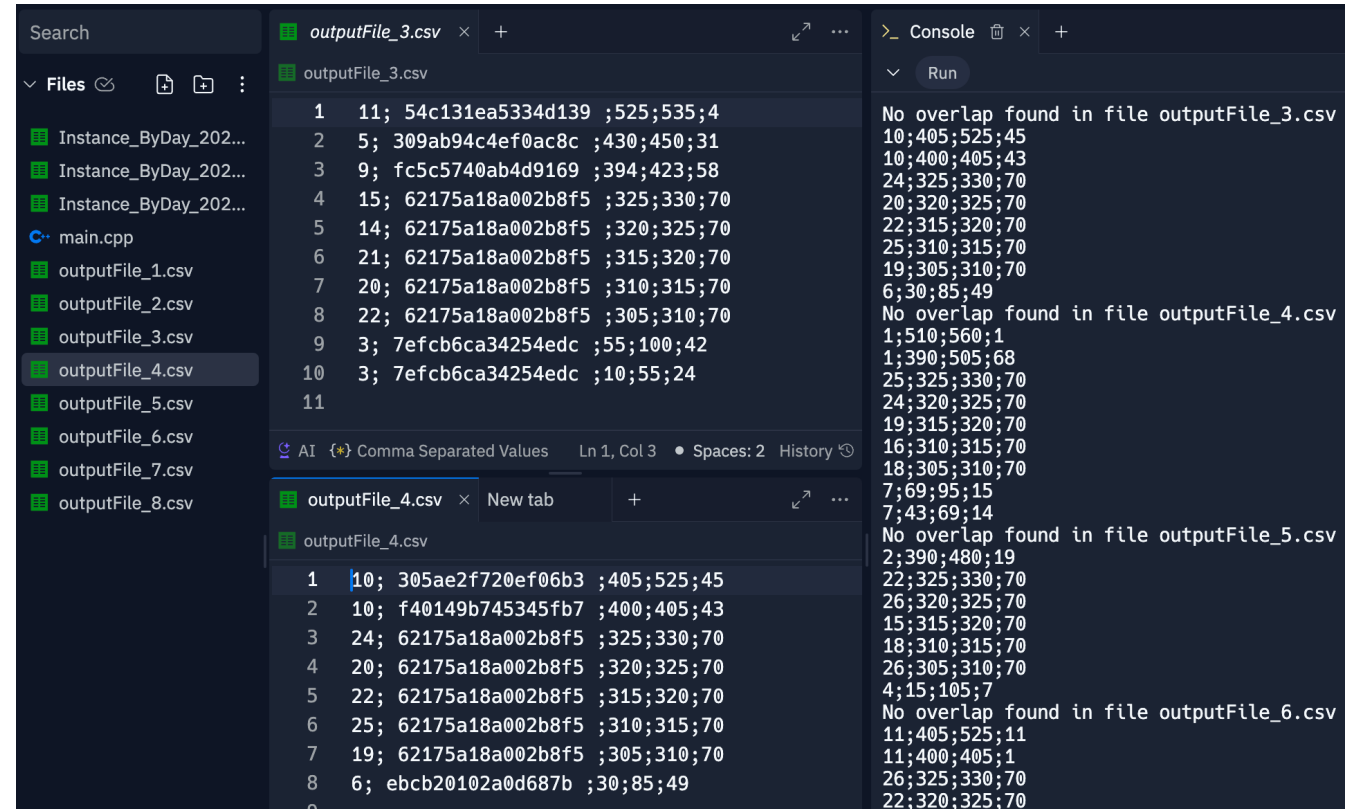
- In fact, we have to run the algorithm once .Then we have to remove the selected intervals
- after that then we re-run the algorithm a second time. And so forth, until we execute it a total of 8 times.

Number of cards	Number of reruns	Number of selected programs	Total weight after selection
4	1	19	95
	2	14	81
	3	10	71
	4	8	63
	total	51	310
6	1	19	95
	2	14	81
	3	10	71
	4	8	63
	5	9	54
	6	7	47
	total	67	411
8	1	19	95
	2	14	81
	3	10	71
	4	8	63
	5	9	54
	6	7	47
	7	7	40
	8	6	34
	total	80	485

Results of the Kleinberg Algorithm with Different Number of Cards

The algorithm to optimize the scheduling (4/4)

- we created a checker (check overlap)
 - The function can open the output_*. csv file, read them and check that an interval does not appears twice times.
- Through this iterative approach,
 - we ensure efficient utilization of all available resources
 - optimize the recording process across multiple machines



The screenshot shows a code editor with a file explorer on the left, a main editor area with two CSV files open, and a console on the right. The file explorer lists files like Instance_ByDay_202..., main.cpp, and outputFile_1.csv through outputFile_8.csv. The main editor shows 'outputFile_3.csv' and 'outputFile_4.csv'. The console displays the output of a checker function, which reports 'No overlap found' for each file and lists the intervals found.

```
Search
Files
Instance_ByDay_202...
Instance_ByDay_202...
Instance_ByDay_202...
main.cpp
outputFile_1.csv
outputFile_2.csv
outputFile_3.csv
outputFile_4.csv
outputFile_5.csv
outputFile_6.csv
outputFile_7.csv
outputFile_8.csv

outputFile_3.csv
1 11; 54c131ea5334d139 ;525;535;4
2 5; 309ab94c4ef0ac8c ;430;450;31
3 9; fc5c5740ab4d9169 ;394;423;58
4 15; 62175a18a002b8f5 ;325;330;70
5 14; 62175a18a002b8f5 ;320;325;70
6 21; 62175a18a002b8f5 ;315;320;70
7 20; 62175a18a002b8f5 ;310;315;70
8 22; 62175a18a002b8f5 ;305;310;70
9 3; 7efcb6ca34254edc ;55;100;42
10 3; 7efcb6ca34254edc ;10;55;24
11

outputFile_4.csv
1 10; 305ae2f720ef06b3 ;405;525;45
2 10; f40149b745345fb7 ;400;405;43
3 24; 62175a18a002b8f5 ;325;330;70
4 20; 62175a18a002b8f5 ;320;325;70
5 22; 62175a18a002b8f5 ;315;320;70
6 25; 62175a18a002b8f5 ;310;315;70
7 19; 62175a18a002b8f5 ;305;310;70
8 6; ebc20102a0d687b ;30;85;49

Console
Run
No overlap found in file outputFile_3.csv
10;405;525;45
10;400;405;43
24;325;330;70
20;320;325;70
22;315;320;70
25;310;315;70
19;305;310;70
6;30;85;49
No overlap found in file outputFile_4.csv
1;510;560;1
1;390;505;68
25;325;330;70
24;320;325;70
19;315;320;70
16;310;315;70
18;305;310;70
7;69;95;15
7;43;69;14
No overlap found in file outputFile_5.csv
2;390;480;19
22;325;330;70
26;320;325;70
15;315;320;70
18;310;315;70
26;305;310;70
4;15;105;7
No overlap found in file outputFile_6.csv
11;405;525;11
11;400;405;1
26;325;330;70
22;320;325;70
```

Results of checking overlap in the output

Conclusions and perspectives

➤ Conclusions:

- We have developed a program auto switch channel
- We have applied Kleinberg algorithm to optimize the interest of the recorded videos.

➤ Further Work

- Improve the Kleinberg Algorithm Performance
- Develop a stable video capturing program with automatic channel switching

THANK FOR YOUR ATTENTIONS !

